下载APP

05 | 图像处理库：如何实现长图拼接？

2021-02-20 尹会生

Python自动化办公实战课 进入课程 >

讲述：尹会生
时长 14:45 大小 13.51M

你好，我是尹会生。

我们在工作中，除了和文字、表格打交道之外，还会经常涉及到批量处理图片和视频的工作。比如：媒体从业者在发微博长图文时，需要把多个图片拼接成一幅长图；作为视频剪辑人员，需要从互联网下载多段视频，再进行合并。

这类工作可以用功能强大的商业软件实现，不过这些软件大都操作繁琐，而且还需要付费。为了降低学习成本和购买软件的成本，我们往往还会使用开源软件替代商业软件来实现图片和视频处理功能。但是开源软件通常都是以命令行方式运行的，所以我们不仅要记住命令，还得记住命令的常用参数。

不过，幸运的是，虽然直接使用开源软件不够友好，但如果通过 Python 来调用这些开源软件，那实现长图和视频拼接就轻而易举了，而且还能大批量地处理图片和视频。

Python 是如何调用外部命令的

为了让你了解 Python 是如何操作这些开源软件的，我先来给你介绍一下 Python 调用外部程序的原理。

我们要想使用 Python 语言之外的功能，要依靠两大途径：导入函数库和调用外部命令。

在第一讲我使用的 xlrd 库是通过 import xlrd 命令导入到 Python 语言中的，Python 语言默认是不支持 Excel 的。那么通过导入函数库，Python 就可以获得对 Excel 的操作能力。

还有一种情况是，需要操作 Python 语言之外的功能，但这个功能没有人将它开发成函数库，那如果我们想要使用这些功能，使用的途径就是调用外部命令了，而调用外部命令就需要 Python 内部函数库的 subprocess 模块来实现。

这个模块的实现机制是：它的 run() 函数的参数可以指定一个可以运行的程序的路径，而 Python 会根据这个路径来运行可执行文件，然后再根据运行结果，以及 Python 的逻辑判断去进行后续的自动化处理工作。

这个实现机制并不难，我给你写一段简单的程序，帮你理解 Python 是怎样调用外部命令的。这里以 macOS 系统为例，我们通过 Python 获取当前目录下所有文件的功能。

复制代码

```
from subprocess import run, Popen, PIPE

cmd1 = ["ls", "."]
returncode = run(cmd1)

print(returncode)
# CompletedProcess(args=['ls', '.'], returncode=0)
# returncode是“ls .”的退出状态码.
# 通常来说，一个为 0 的退出码表示进程运行正常

# 使用Popen获取程序运行结果
with Popen(cmd1, shell=True, stdout=PIPE, stderr=PIPE, encoding="utf-8") as fs

```

```
    # 如果程序在 timeout 秒后未执行完成,会抛出 TimeoutExpired 异常
    fs.wait(2)

    # 从标准输出中读取数据,知道文件结束
    files = fs.communicate()[0]

print(files)

```

这段代码中最核心的函数是 run() 函数和 Popen 类。subprocess 模块就是通过这两个函数实现的外部程序调用。我来为你重点剖析一下它们的功能、参数，以及何时选择 run() 函数、何时选择 Popen 类。

为了实现 Python 调用可执行文件，首先在代码的第一行，我是这样编写的：

复制代码

```
from subprocess import run, Popen, PIPE
```

这样一行代码，它和我第一讲使用的 import 方式导入函数库的区别是，这种形式可以让你直接使用模块中的类和方法。

如果你使用 “import subprocess”方式导入 subprocess 库的话，在调用 run() 函数的时候，就需要用 “库. 函数”的形式在 Python 中使用库当中的函数，
即“subprocess.run()” 。在你多次调用 run() 函数时，代码会较长，那么使用“from import”方式导入，就可以在当前代码文件中直接使用 run() 函数，为代码的阅读带来更好的体验。

接下来，我定义了一个变量 cmd1。这个变量的值是 macOS 命令行能够运行的“ls .”命令，这个命令的执行结果是显示当前目录下所有文件和文件夹的名称。

run() 函数的主要功能就是执行一个新的程序，它的用法非常简单，把第一个参数指定为要执行程序的路径就可以了。如果要执行的程序带有参数，那就可以使用列表数据类型存放可执行程序名称和参数，像是我在程序中定义的 cmd1 变量一样。如果你需要运行其他命令，把代码中的 ls 替换为你想要运行的其他程序就行了。

为了让 Python 自动化处理程序更强大，除了运行程序外，你还可以得到可执行程序的运行结果。在这种情况下，我们就需要使用 Popen 类替代 run() 函数实现外部程序的调用。

可以看到，我在代码的第 12 行先通过 Popen 类执行了“ls .”命令，接着通过参数 stdout=PIPE 将命令的执行结果放入到 PIPE 对象中，最后再通过 communicate() 函数将 PIPE 中的内容读取出来，存放到 files 变量中，这样就实现了读取命令执行结果的功能。

这个功能是无法在 run() 函数实现的，因此在你需要通过 Python 读取程序执行结果的时候，就可以选择 Popen 类。不过如果只需要运行可执行程序，那使用 run() 函数就能满足你的要求了。如果你想更深入地了解它们，我建议你阅读 subprocess 库的官方文档。

以上就是我用 subprocess 库实现 Python 调用可执行程序的方法。Python 之所以被我们称作最佳的“胶水语言”，就是因为它能轻易“粘合”可执行程序。利用 Python 灵活的逻辑判断、循环语法可以实现程序的批量执行和流程管理。

接下来，我们就使用 subprocess 来实现长图拼接和视频拼接的功能。

长图拼接

当我进行微博文案推广的时候，需要将多个图片拼接成一个长图。拼接图片的功能 Python 本身是不具备的，因此就需要引入外部命令来实现图片拼接功能。

我在 macOS 平台上找到了一个非常强大的图像处理软件叫做 **ImageMagick**，它能对图片进行编辑、合并、切割、旋转等 90 多种操作。 ImageMagick 软件实现图片拼接的命令格式是这样的：

```
1 composite 图片1.jpg 图片2.jpg ... 图片n.jpg 最终合成结果.jpg
```

复制代码

在这段命令格式中，composite 命令的参数包含了多个图片文件，每个图片需要对照着文件将图片的路径和文件名写在参数中。如果手工输入图片名称，不仅效率低，而且容易遗漏。另外，如果需要大量重复使用 composite，还需要精细调整合并结果，给 composite 程序增加很多参数。

因此，我就可以通过 Python 调用可执行程序的 subprocess 库，对 composite 拼长图的工作进行脚本化编程。它的核心实现代码如下：

复制代码

```
p = Path(jpg_path)

# 增加命令
cmd = ["composite",]

# 增加参数
for x in p.iterdir() if  PurePath(x).match('*.jpg'):
    cmd.append(x)

# 增加结果
cmd.append(result_path)

run(cmd)
```

由于 composite 可以把长图合成的结果直接输出为文件，因此采用 run() 函数即可实现程序执行的功能。另外，当你需要调整 composite 参数时，可以直接修改 cmd 变量的值，并不需要改动程序其他部分。当你要对新的一组图片进行合成的时候，重新设置 jpg_path 变量就行了。

总结来说，使用 Python 调用 composite 合并的好处就是：你不用记住程序使用的繁杂的命令行参数，也不用记住运行逻辑，因为 Python 程序已经事先把逻辑编写好了。

视频的拆分与合并

在了解了如何使用 subprocess 调用 composite 实现长图拼接之后，我再给你讲一下如何使用 subprocess 库调用可执行程序，来进行视频的拆分与合并。

我们先来学习下视频拆分的原理。

你在电脑本地经常见到的视频格式是 MP4，但如果要把视频放在互联网上，为了减少首次播放的加载时间，你就必须把一个 MP4 切分成多个文件，而且切分之后还需要把格式转换为.TS 格式的视频文件。

为什么不直接使用 MP4 格式，而是要把 MP4 格式改成.TS 格式呢？这是因为.TS 格式可以保证多个文件之间的视频无缝播放，而且还会保证视频不会在播放下一个文件的时候，出现破音或画面中断等影响用户体验的情况。

当我们将一个视频切分成多个文件的时候，就要考虑文件的播放顺序问题了。为了记录顺序，我们需要在切分之后引入一个索引文件，这个索引文件不用手动编写，我们直接使 FFmpeg 命令就行了，它可以实现视频格式的转换、合并和拆分。FFmpeg 命令会在切分之后，自动产生一个以.M3U8 结尾的索引文件。

我来解释一下这个索引文件。M3U8 文件是指 UDF-8 编码格式下的 M3U 视频索引，播放器通过这个索引文件就可以找到视频下所有的分段，并依次播放视频。

看到这儿你应该就能明白了，想要使用 Python 进行视频拆分，我们首先需要 FFmpeg 命令，然后通过 Python 设置 FFmpeg 的参数，最后再指定 MP4 文件和.TS 文件的路径，这样就能实现拆分视频的功能了。因此我使用这样的代码来实现视频拆分：

复制代码

```
from subprocess import run
input_video = "/Users/edz/Desktop/05/xxx.mp4"
segment_time = 10
m3u8_list = "/Users/edz/Desktop/05/xxx.m3u8"
output_video = "/Users/edz/Desktop/05/video-%04d.ts"

cmd1 = ["ffmpeg", "-i", input_video, "-f", "segment", "-segment_time", str(seg
    "mpegts", "-segment_list", m3u8_list, "-c", "copy", "-bsf:v", "h264_mp4toa

run(cmd1)

```

在代码中，我通过 FFmpeg 把 MP4 切分成了多段 TS 文件。你要想实现相同功能，首先需要在电脑中安装 FFmpeg 命令，它的下载地址为：https://ffmpeg.org/download.html。

为了实现 MP4 文件格式的分割，需要使用 ffmpeg 非常多的参数。不过使用 Python 进行调用的好处，就是你不用记住复杂的参数。我们把输入文件路径、切分大小、输出的 M3U8 和 TS 文件指定为四个变量，这样只修改这四个变量，就可以实现拆分功能了。

如果你需要离线观看视频，就要将网络上的视频下载到本地，这时你会发现从互联网下载的格式是 M3U8 和 TS 文件。那又怎么把它们合并成 MP4 文件呢？

你同样可以使用 FFmpeg 命令，但是 FFmpeg 的参数不同。我将 FFmpeg 的命令写在这里：

```
ffmpeg -allowed_extensions ALL -protocol_whitelist "file,http,crypto,tcp,https
```

如果你不想背诵这么长的参数，完全可以仿照 Python 整合拆分视频的代码来实现合并功能。先 FFmpeg 命令和参数放入列表，再把 M3U8 文件和 MP4 文件放入变量，便于你合并新的视频的时候进行重新赋值。

所以你看，相比直接使用 FFmpeg，subprocess 调用 FFmpeg 的优势就在于两点，一是不用记住复杂参数，二是对批量转换视频非常有利。举两个例子。

如果你是视频剪辑的专业工作者，肯定要大量使用 FFmpeg 更复杂的功能，这些功能对应的参数一般都比较多，而且参数很多都使用了简写和大小写，很难记忆。但要是使用 Python 调用的话，你可以直接更改要操作的文件路径，就不必记录大量的参数。

另外需要进行视频的批量转换时，可以通过第一讲的循环操作对视频任务批量处理，这样就避免了手动逐个修改书写文件的操作，从而提高视频转换的效率。

小结

最后，我来为你总结一下这节课的主要内容。

通过对 subprocess 库的讲解，你知道了怎样通过它实现 Python 加载外部可执行程序，并且能够对程序执行的结果进行处理。

我也为你讲解了长图拼接和视频拆分合并的两个案例，帮你更好地理解 Python 为什么会被称作“胶水”语言。

我还想强调一下，通过 Python 调用可执行程序的用法非常常见，特别是在多媒体处理、自然科学、AI 等领域里。在这些专业领域，为了加快计算速度，通常会使用 C++ 语言实现专业程序。

这些专业程序参数多、功能单一，且使用命令行执行，当你需要多次执行这些程序，又不想背诵它们的参数的时候，就可以利用 Python 的判断循环功能，结合 C++ 语言实现的专业程序，来实现批量执行和减少参数手动输入的工作，提高你的工作效率。

最后，我也把这节课的代码附上，你可以查看。本讲代码

思考题

在最后也请你思考一下，你在工作当中是否会使用命令行工具呢？它们能否用 Python 进行包装，从而避免手写复杂参数呢？

如果你觉得这节课有用，能解决你的办公效率问题，欢迎你点击“请朋友读”，分享给你的朋友或同事。

提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 04 | 函数与字典：如何实现多次替换

下一篇 06 | jieba分词：如何基于感情色彩进行单词数量统计？

精选留言 (6)

写留言

Sicolas Flamel
2021-02-20

弱弱的问一句Windows用户可以不?

展开

作者回复: Windows是不行的哦，这个软件是 Linux和MAC下边实现 photoshop的一个工具，主打的就是通过命令行进行批量处理。在windows下想实现这种功能，可以使用photoshop的“动作”来完成。

2 2

SWC603

2021-03-20

老師，是否可以利用python 一次過批量更改excel中的公司logo? 因原有excel 檔中公司logo更改了新設計。 謝謝你!

展开

作者回复: 你好，对于excel中的图片处理，步骤比较多，但是操作不难。
首先你需要将excel文件的扩展名改成zip格式，
然后对zip格式进行解压缩
利于我在后面讲解的遍历文件的方式找到图片，并替换成新的图片
最后，再将文件夹打包回zip并重命名为xlsx

赤い悪魔
2021-03-08

MacOS
cmd1 = ["ls", "."]
returncode = run(cmd1)
报错，替换了
returncode = run('ls', '.', shell=True)

展开

作者回复: 我解释一下报错的原因，是因为ls命令在运行的时候需要环境变量，而环境变量是存在于shell中的。所以需要增加shell=True。

xiaocao
2021-02-21

另外问一下：外部命令还可以调用哪些?如何查找和学习?

作者回复: 外部命令功能，可以调用任何命令行执行的命令，最常见的就是文件操作类命令， windows可以通过cmd命令作为关键字搜索，mac可以通过shell脚本作为关键字搜索到

梁健
2021-02-20

我客观地说个存在的问题，您使用Mac OS的操作系统下的 ImageMagick进行教学，这给

我们这些windows用户带来很多的不方便，命令行执行上这两个系统的命令完全不一样（哭了），我找了好多教程都没找到好的解决办法。希望后面的内容还是能够多在Windows上做。ps：今天这课是学的到现在最烦燥的一节（痛彻心扉）。

展开

作者回复: 先不要痛彻心扉，我在这节课主要想把调用外部命令的方式交给你，windows下也有很多软件支持命令行的，最常见的就是文件和文件夹的操作命令。你可以通过搜索引擎搜索windows 命令行自带的命令，来找到他们。在我工作的早期，接触的服务器大部分就是windows的操作系统，我需要进行定时任务和各种复杂任务的编排的时候，都会通过cmd命令行的命令来完成。有了Python进行封装之后，这些命令肯定更好用了。你可以去了解一下。

Soul of the Drago...

2021-02-20

同问，请问上面的代码仅适用于macOS操作系统吗?

展开